AD-A008 152

A HEURISTIC APPROACH TO COMPUTER SYSTEMS PERFORMANCE IMPROVEMENT, I: A FAST PREDICTION TOOL

Stephen R. Kimbleton

University of Southern California

Prepared for:

Office of Naval Research
Advanced Research Projects Agency

March 1975

/

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE	READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER ISI/RR-74-20 — 2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER AD/A 00 8152
4. TITLE (and Subtitle) A HEURISTIC APPROACH TO COMPUTER SYSTEMS PERFORMANCE IMPROVEMENT, I: A FAST PREDICTION TOOL	5. TYPE OF REPORT & PERIOD COVERED Technical
	6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) Stephen R. Kimbleton	8. CONTRACT OR GRANT NUMBER(s) DAHC 15 72 C 0308 and N00014-67-A-0181-0036
9. PERFORMING ORGANIZATION NAME AND ADDRESS USC Information Sciences Institute 4676 Admiralty Way Marina del Rey, California 90291	10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS ARPA Order #2223/1 ONR #NR 049-311
11. CONTROLLING OFFICE NAME AND ADDRESS Advanced Research Projects Agency 1400 Wilson Boulevard Arlington, Virginia 22209 (OVER)	12. REPORT DATE March 1975
	13. NUMBER OF PAGES 32
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)	15. SECURITY CLASS. (of this report)
	15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

This document approved for public release and sale; distribution is unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18 SUPPLEMENTARY NOTES

A condensed version of this report will appear in the Proceedings of the 1975 National Computer Conference.

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Performance prediction, simulation, analytic models, system design, performance analysis

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

The complexity of current computer systems strongly inhibits management usage of analytic modeling techniques in seeking 'good' computer system performance. As a result, heuristically-based approaches are in order. Implementation of such an approach requires: (1) a means for determining the performance of a given computer system processing a given collection of jobs in accord with a specified schedule, (2) a means for achieving an improved system from a given system, and (3) a technique for determining when to stop this iterative process. The objective of this paper is to describe an analytically driven

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE
S/N 0102-014-6601

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

SECURITY CLASSIFICATION OF THIS PAGE(When Data Entered)

11. Office of Naval Research
Department of the Navy
Arlington, Virginia 22217

20.
approach to computer system performance prediction which can achieve execution speeds of approximately two orders of magnitude faster than real time for production batch installations while providing detailed information on device utilizations and delays. Thus, it provides a basic required capability for the development of heuristic approaches to computer system performance improvement.

SECURITY CLASSIFICATION OF THIS PAGE(When Data Entered)

ARPA ORDER NO. 2223

ISI/RR-74-20

March 1975

Stephen R. Kimbleton

A Heuristic Approach to Computer Systems Performance Improvement, I: A Fast Performance Prediction Tool

INFORMATION SCIENCES INSTITUTE

UNIVERSITY OF SOUTHERN CALIFORNIA

4676 Admiralty Way/Marina del Rey/California 90291

(213) 822-1511

Preparation of this paper was supported by the Advanced Research Projects Agency under Contract No. DAHC15 72 C 0308, ARPA Order No. 2223, Program Code No. 3D30 and 3P10, and the Office of Naval Research, Information Systems Program under Contract N00014-67-A-0181-0035 (NR 049-311).

ABSTRACT

The complexity of current computer systems strongly inhibits management usage of analytic modeling techniques in seeking 'good' computer system performance. As a result, heuristically-based approaches are in order. Implementation of such an approach requires: (1) a means for determining the performance of a given computer system processing a given collection of jobs in accord with a specified schedule, (2) a means for achieving an improved system from a given system, and (3) a technique for determining when to stop this iterative process. The objective of this paper is to describe an analytically driven approach to computer system performance prediction which can achieve execution speeds of approximately two orders of magnitude faster than real time for production batch installations while providing detailed information on device utilizations and delays. Thus, it provides a basic required capability for the development of heuristic approaches to computer system performance improvement.

KEY WORDS: Performance prediction, simulation, analytic models, system design, performance analysis

CR CATEGORIES: 2.44 3.50 4.30 5.5

ACKNOWLEDGMENTS

The author would like to express his appreciation to Dr. Thomas E. Bell who critically reviewed an earlier draft of this paper, to Mr. James Cochran who transferred ASIM to the Institute TENEX computer system and executed all experimental runs, to Mr. Nelson Lucas who prepared the figures, and to Mrs. Linda Tisnado who prepared the report. A condensed version of this report will appear in the Proceedings of the 1975 National Computer Conference.

INTRODUCTION

Satisfaction of the organizational information processing requirement can be achieved through an iterative process involving requirements specification and system design entailed by a given specification. Because of its obvious importance, requirements specification has received considerable study (an excellent survey is provided in [COUGJ 73]) and semi-automated procedures for checking consistency and completeness [TEICD 71] have been developed.

Computer system design for a given requirements specification has also proven to be a difficult problem for: vendors, the organizational information processing function, the computing center manager and the user. Much of the available literature concerned with this topic is found under the general classification of performance analysis and much of this literature in turn, is directed towards the vendor. In part, this focus appears to reflect the substantial leverage implicit in the development of vendor oriented performance tools.

Development of computer system design procedures appropriate to the computer center manager also appears to have significant leverage. This topic can be structured into three components: vendor selection, computer system sizing, and computer system tuning [LUCAH 71]. Since vendor selection is substantially dependent upon considerations other than minimum cost determination of a system appropriate to stipulated requirements (such as maintenance, compatibility, etc.), our usage of the term system design will be in the context of sizing and tuning. Clearly, methodologies which improve sizing and tuning capabilities provide expanded support for vendor selection.

Computer system design methodologies oriented to the computer center manager are required for two purposes: (1) determination of the reasonableness of existing performance, i.e., 'glitch' detection, and (2) evaluation of the effects of system modifications and alterations. Although vendor oriented results are of general interest, very tight constraints are implicit in the requirement for determining the performance of a given system executing a specified workload in accord with a clearly defined schedule. This constraint mitigates against the utility of a parametric (subsystem) approach and argues for the development of a coherent, coordinated (system) approach integrating data gathering, data transformation (including performance prediction), and information display and archival.

Development of a system design methodology based on algorithmic approaches is precluded by the complexity of the computing center environment. Instead, development of heuristic (informed trial and error) system design approaches is required. The ability of heuristic approaches to effectively solve complex problems is apparent from the quality of current computer chess playing programs. Efficiency must also be considered in investigating systems design.

Heuristic System Design Requirements

Development of a heuristic approach to computer system design requires three capabilities:

1. Performance Prediction to determine the exact performance of a given system processing a specified workload in accord with a given schedule.

2. System Comparison to determine the more desirable of two system alternatives in the context of a given schedule and workload.

3. Stopping Time Determination for terminating the iteration cycle implicitly defined by (1) and (2).

Achievement of the first capability requires a very fast performance prediction tool; its absence appears to be a major reason underlying the current lack of heuristic approaches to system design. The second capability is implicitly provided by the output information generated through performance prediction together with any summary transformations that may be employed. One possible comparison criterion is provided in [KIMBS 74B] for use in scheduling batch computer systems. Precise specification of this information requires identification of the decision categories available to computer center management; a first order structure is given below. Investigation of stopping time mechanisms can safely be deferred in view of the exploratory nature of heuristic approaches.

The preceding comments imply that development of a fast performance prediction mechanism is the major requirement for development of a Heuristic System Design (HSD) methodology.

Computer Center Management Decision Structure

Development of a methodology to support computer center management require classification of the major decisions. This classification results in two categories: organizational and technological issues. Some representative issues are provided in Table I.

TABLE I

ISSUES IN COMPUTER SYSTEMS PERFORMANCE

Organizational Issues	*Technological Issues*
Unnecessary jobs	Hardware configuration
Programmer training	Software support
Priority structure requirements	Job schedules
Online support requirements	Device schedules
Fractional system capacity for interactive support	File-Device assignment
Local control of data base	Device/Channel ratio
System compatibility requirements	Extended instruction set
Control limit specification	Processor scheduling algorithm

First order technological decisions can be divided into four classes:

D1. hardware alterations and enhancements,

D2. software capabilities,

D3. job schedules, and

D4. file-device assignments.

For technological issues the concept of an optimal system is reasonable although elusive.

Organizational issues have two significant characteristics: (1) management satisfices rather than optimizes on these variables, and (2) determination of reasonable values for these variables is heavily site dependent. Thus, such issues cannot be resolved through a generic approach and must consequently be attacked on a case by case basis in which the skills of the management consultant are often required.

The distinction between organizational and technological issues is not inviolate. For instance, operator efficiency, an organizational issue, is significantly affected by the technological support provided (advance notification of tape mounts, scheduling assistance, etc.).

Decisionmaking is further affected through the presence of management and user constraints. Management constraints are explicitly represented through budgetary limitations which implicitly reflect a desire to assure a 'reasonable' level of system utilization. Presumably, utilizations of 100% on all devices would be regarded as completely acceptable. User constraints are reflected in the acceptable range of response/turnaround times and times equivalent to those experienced on a stand-alone system would be regarded as completely acceptable.

A critical function of computer center management is achievement of an acceptable system providing reasonable balance among these constraints. For a given requirements specification, this constitutes the system design problem and its two components, sizing and tuning.

Paper Objectives

The thesis of this report is that a support methodology appropriate to computer center management can be developed, that this development requires a heuristic approach, and that the major implementational requirement for such an approach is the development of a fast performance prediction capability. To support this thesis we have provided an overview of the computer center management decision process and identified the major components of a heuristic system design approach. Next, we shall identify performance prediction requirements implicit in this approach, and then describe one means for their realization. Comparisons with a computer system simulator are provided and observations regarding desirable directions for future research are made.

PERFORMANCE PREDICTION REQUIREMENTS

Development of an HSD supportive performance prediction mechanism requires identification of: (1) objectives and information requirements, (2) accuracy requirements, (3) level of detail, and (4) modeling technology. In this section, goals for each of these categories will be detailed. The remainder of the paper may be viewed as a demonstration of the feasibility of their achievement.

Objectives and Information Requirements

Technological control of a computer system requires balancing management utilization objectives with user response/turnaround time goals. This requires determination of delay by job class and identification of multiprogramming related delays. (A turnaround time of ten hours is probably acceptable if the stand-alone processing time is nine hours and is probably unacceptable if the stand-alone processing time is ten minutes.)

Technological control decisions are reflected in the four decision categories (D1)-(D4) cited above. It follows that evaluation of alternatives requires knowledge of individual device utilizations and delays. Further, it is desirable that this information be broken out on a per job basis, a per shift basis, and as will be seen later, a per time segment (period of time during which the composition of the mix remains constant) basis.

Accuracy Requirements

Persuasive salesmanship has been used to convince computer center management that accuracy levels of one percent are readily obtainable with current, commercially available simulators. Further, feasibility demonstrations show that such accuracy levels can be achieved after some tuning of the simulator. The existence of this tuning requirement mitigates against the utility of the simulator in investigating new configurations or in evaluating the reasonableness of existing performance--two primary uses for a simulator.

The difficulty of designing and implementing a simulator is a function of the accuracy level required. The user of a simulator naturally seeks a very high accuracy level, say 1% (that is $|obs - pred|/obs < .01$ where obs denotes the observed value of a statistic and pred denotes the value of the statistic predicted through usage of the simulator). Acceptance of a lower level of accuracy requires education of the user through identification of accrued advantages typically including: reduction/elimination of the need for calibration, increased speed of execution, and expedition of the verification/validation process. Based upon discussions with managers, it is the author's conjecture that most users can easily tolerate an accuracy level of 20%. This paper hypothesizes, and partially verifies, that such an accuracy level can be achieved through usage of fast, analytically driven techniques.

Level of Detail

Performance prediction techniques can be classified in terms of the basic level of detail incorporated: hardware (register), operating system, resource allocation, time segment and job. Decreasing the level of detail decreases the potentially achievable level of accuracy as well as the cost of implementation and cost of verification/validation of the simulation. Our objective is to achieve an accuracy level of twenty percent, and our assertion is that this level of accuracy can be achieved through prediction of time segment performance and aggregation of the resulting statistics to achieve job and shift performance. This assertion is unprovable. However, supporting evidence is provided in the comparisons detailed in Figures 1-5, included at the end of the paper. Although the 20% discrepancy level is not achieved in all cases, non-achievement seems clearly due to usage of an approximation technique providing only lower bound estimators for processor utilization. Subsequent efforts directed to achievement of the desired tolerance level through development of improved processor utilization estimators seem very likely to succeed. Moreover, it will be noted that the level of discrepancy decreases as the level of processor utilization increases.

Modeling Technology

Prediction of system performance can be approached through either analytic or simulation techniques. Because of the limited information available from an individual analytic tool, analytic approaches have been relegated to the role of design tools used in gaining insight into subsystem tradeoffs and policy. Additionally, analytic models are occasionally used to estimate gross system performance characteristics in which extensive aggregation of system components is required to achieve computational feasibility [MOORC 71], [HANSF 71]. The advantage of analytic approaches is their speed of implementation and execution coupled with relatively low cost for developed models; the disadvantages include extensive simplifying assumptions, the feasibility of incorporating only a meager level of detail, the difficulty of understanding them without extensive training in modeling techniques, and the prevailing lack of reasonable user interfaces.

Simulation approaches, in contrast, permit incorporation of almost any desired level of detail. Since there is a prevailing tendency to incorporate superfluous factors, and a great discrepancy between the rate at which events occur in a computer system (micro seconds--milli seconds) and the run time of a computer system (hours--days), most simulation models execute relatively slowly, e.g., 1-10 times faster than real-time.

Analytic and simulation approaches can be regarded as two endpoints of a spectrum of possible approaches to computer system performance prediction. Intermediate points within this spectrum would blend simulation and analytic approaches in a desire to balance execution speed, output information, required input data, and accuracy. The objective of this paper is to demonstrate that through proper development of an analytically driven approach, information essentially equivalent to that provided through current, commercially available simulators, can be achieved very quickly. Thus, the basic requirement for implementation of a heuristic approach to computer system design, sizing, and tuning will have been achieved.

ANALYTIC PREDICTION OF SYSTEM PERFORMANCE

Development of an analytic technique for predicting computer system performance requires identification of: (1) the precise analytic techniques to be used, (2) the input data required, (3) the output information sought, and (4) the techniques to be used to derive (3) from (2). Item (4) proves not to be straightforward since a variety of analytic techniques must be interrelated to obtain a capability which can provide output information competitive with that provided by a resource allocation level simulator. Each of these issues will now be discussed in turn.

Selection of Basic Analytic Approach

Two major approaches to modeling processor-I/O interaction within a computer system can be differentiated through their representation of I/O delay. The queuing network approach represents I/O devices by their service time and obtains global system results (device utilizations and delays) in terms of these service times and the transition matrix guiding migration among I/O devices and the processor (also represented in terms of its service time). Alternative analytic approaches represent I/O devices in terms of total I/O delay and are provided by the machine repair model and an approximation technique termed the system process model [KIMBS 75].

Wide usage of the queuing network approach has been made in computer network analysis [KLEIL 64], communications system design [KLEIL 70], and computer systems analysis [BASKF 71], [BUZEJ 71] and [MOORC 71]. The results obtained via this approach have proved encouraging and extensive research is continuing on a variety of related topics. Usage of this approach as the kernel of an analytically driven performance prediction mechanism proves difficult for two reasons: (1) representation of data path delays, and (2) computational complexity. The first reason reflects the fact that incorporation of data paths requires a mathematical capability to handle queuing networks with correlated service times - a capability which currently appears to be in advance of the state of the art. The second reason reflects the fact that the complexity of the queuing network approach tends to increase with the square of the number of nodes incorporated. Thus, published studies have found lumping of nodes necessary to reduce the dimensionality from the fifty or more resources comprising the typical large scale computer system to 5-10 [MOORC 71]. However, some work has appeared which is concerned with more computationally efficient procedures [BUZEJ 73].

In I/O delay approaches, the input information is mean processor burst and mean I/O delay; the output information is processor utilization and delay. The standard machine repair model assumes exponentially distributed variables and an iterative technique has been developed [GAVED 67] for calculating the relevant statistics under the assumption of non-exponentially distributed I/O delays. This technique has been successfully applied to computer system modeling in which the workload is characterized by an 'average' job(s) [SEKIA 72].

The system process model [KIMBS 75] uses a similar job characterization. Exponential assumptions are shown to constitute a natural midpoint for a wide range of distributional assumptions. The estimator of processor delay is independent of the approximation used to obtain processor utilization; thus, alternative techniques for estimating processor utilization, including the machine repair approach, can be used. This computational technique has been used because of the straightforward nature of the calculations which constitute the kernel computation used in developing an analytically driven performance prediction technique termed ASIM.

I/O delay approaches naturally enable a hierarchical approach to calculation of system statistics in which I/O delay is first determined followed by evaluation of other system performance characteristics. Since, as shown in the following section, the kernel calculation for I/O delay and processor utilization is relatively straightforward, the complexity of the total computation is approximately linear in the number of devices comprising the system, and is thus competitive with simulation approaches for both large and small systems.

DESCRIPTION OF AN ANALYTICALLY DRIVEN PERFORMANCE PREDICTION TOOL (ASIM)

The major objective in the development of ASIM was fast prediction of the performance of a given system executing a specified collection of jobs in accord with a defined schedule. in this section we describe: (1) simplifying assumptions used in the initial ASIM implementation, (2) input data characterization, (3) output information developed, (4) conceptual steps involved in obtaining system performance statistics, and (5) accuracy/timing considerations.

ASIM Implementation Assumptions

The following eight simplifying assumptions were chosen to represent a compromise between code simplification and demonstrating feasibility appropriate to actual utilization in a computing environment.

1. A single processor system.
2. One data path per device.
3. Zero setup times for jobs (both initiation/termination and disk/tape mount/dismount times).
4. All jobs have equal priority.
5. One active data set per device per job.
6. No unit record equipment.
7. Jobs use a fixed amount of user memory.
8. A maximum of thirty drums, disks, and tapes; all drums are shared, a user specified number of disks may be shared, and all tapes are non-shared.

Removal of the single processor assumption is possible at the relatively minor cost of requiring another (internal) list for each additional processor and incorporation of an estimator for processor contention impact on system performance. Explicit incorporation of data paths requires development of a suitable approach to representation of data path induced delay; existing models in the literature are oriented to static prediction and inappropriate to the objectives of this model. Removal of the remaining assumptions is a programming exercise (note that average working set size can be used for virtual memory systems in place of partition size for those systems based on a working set philosophy).

ASIM Input Data

The input data to ASIM consists of three files: (1) Job/System Descriptions, (2) System Characteristics, and (3) Switch Settings. Switch settings constitute a superset of the standard GASP switch settings [PRITA 69] which initialize a run and will not be discussed further.

Table II indicates the components of a Job/System Description (JSD). As is apparent, a JSD is related to a synthetic module characterization of a job [BUCHW 69], [HAMIP 73], [SREEK 74] and consists of two components. The exogenous component describes the environment defined for the job and its relation to other jobs, while the endogenous component describes the resource requirements and utilization rates for an individual job. Although automatic generation of JSD's is not currently feasible, careful examination of the more sophisticated accounting systems demonstrates the feasibility of capturing these data through minor modifications. Indeed, SMF data is currently being utilized to generate input data for some currently available simulators [EDGEG 73].

It should be noted that the JSD was so named since it constitutes a viable description of the job only in the context of a given target system. In particular, no assertion is intended that the same source code executed on systems produced by two different vendors will be the produce the same JSD's.

Jobs are assumed to be initiated in the order indicated by the list of JSD's subject to three constraints: precedence satisfaction, arrival time satisfaction (a job cannot be initiated prior to arrival), and resource availability. If one of these constraints is unsatisfied, initiation is deferred pending its satisfaction. Further, no attempt is made to look ahead in the JSD sequence to determine if another job could be initiated. It should be noted that this requirement is necessary in order to permit study of computer system performance as a function of the schedule [KIMBS 74B]. It does not reflect current job scheduling in which both the operating system and the external scheduling mechanism jointly share responsibility. Although the current system scheduling approach has proven useful (perhaps unavoidable), in practice it greatly impedes determination of the performance of a given computer system executing jobs in accord with a prescribed schedule. Accordingly, the approach which we have described has been implemented. Modification of this approach to more accurately reflect the characteristics of scheduling as implemented on an individual computer system is essentially a programming problem.

TABLE II

JOB/SYSTEM DESCRIPTION (JSD) COMPONENTS

Exogenous JSD Variables

1. Name
2. Time of Arrival
3. Static Priority
4. Job Due Date
5. Job Setup Time
6. Job Precedence Relations
7. Number of Job Steps

Endogenous JSD Variables

1. Processor Interaction
    1. Job Processor Requirements
    2. Processor Burst Description
    3. CPU/I/O Overlap

2. Memory Requirements

3. File List
    1. Number of Files Accessed by Job
    2. File Names
    3. File Access Count
    4. Data Transfer Size

4. File Characteristics
   (For Each Listed File)
    1. File Device Location
    2. Latency
    3. Seek Time Distribution (for disks)
    4. File Size
    5. File Organization

Table III indicates the collection of information comprising the system characteristics. Their determination is straightforward.

TABLE III

SYSTEM CHARACTERISTICS

Number of Drums	0*
Number of Disk Drives	3
Number of Tape Drives	0
Number of Non-Shared Disk Drives	0
Amount of Core Memory	100
Rotation Time of Drum	35.0000
Number of Pages per Drum Track	6
Rotation Time of Disk	16.6667
Number of Pages per Disk Track	5
Total Number of Cylinders	411
Minimum Seek Time	10.0000
Maximum Seek Time	55.0000
Average Seek Time	30.0000
Tape Interrecord Gap	0
Cost Per Unit Time of CPU	10.000000
Cost of Core Per Unit Time	1.000000
Cost of Drum Per Unit Time	0.015000
Cost of Disk Per Unit Time	0.006800
Cost of Tape per Unit Time	0.005000

*Numbers are representative values used in the example.

ASIM Output Information

Output information generated by ASIM is aggregated into three categories: (1) job performance statistics, (2) shift performance statistics, and (3) time segment performance statistics. The information in each category consists of device utilizations and delays as shown in Table IV together with some header information which is not given. For a job this header information includes job identification and for a time segment the collection of jobs in concurrent execution is displayed.

TABLE IV

ASIM OUTPUT INFORMATION

Processor Utilization
Processor Delay

Memory Utilization

Average Degree of Multiprogramming
Average I/O Delay
Average System Utilization

Individual Drum Utilization
Individual Drum Delay
Individual Drum Service Time

Individual Disk Utilization
Individual Disk Delay
Individual Disk Service Time

Individual Tape Utilization
Individual Tape Delay
Individual Tape Service Time

ASIM Performance Calculations

Let an event denote the time at which either a job initiation or job termination occurs and let a time segment denote the time between two successive events. For a shift in which N jobs are processed, there will be at most 2N events and 2N-1 time segments; the upper bound is reached for non-concurrent initiations. Since the composition of the mix is constant over a time segment, analytic prediction of time segment performance is in order. Standard aggregation techniques used for statistics accumulation in simulation can then be used to develop job and shift performance. Additionally, time segment performance is captured since poor performance for a job implies poor performance for at least one time segment. Further, development of heuristic scheduling procedures requires knowledge of time segment performance characteristics [KIMBS 74B] to evaluate the desirability of alternative job interchanges in heuristically seeking a 'good' schedule.

Label the collection of jobs to be processed during a representative time segment J1,...,Jn. Generation of processor delay and utilization requires generation of an average active and blocked interval for this time segment appropriate to this collection of jobs.

Determination of average I/O delay requires care. In its calculation, an average processor burst which is simply the arithmetic average of the processor bursts for jobs J1,...,Jn, is used. It is evident from the Job/System Descriptions that knowledge of the JSD's for J1,...,Jn permits straightforward calculation of the probability of a reference to an individual device during the time segment (assuming accesses are equally probable over a time segment). This probability, together with the average processor burst described above permits a mathematical determination of upper and lower bounds for the interarrival times to a given device. (Determination of the mean interarrival time proves difficult since it is affected by device characteristics and queues; thus, its determination requires an iterative approach and was deferred in the interest of computational speed.) Given these bounds, bounds for device delay can then be obtained through queuing calculations.

Usage of available device models within the literature is a natural objective which is precluded by their assumption of an infinite calling population. In practice, for batch systems, a degree of multiprogramming in the range of 3-6 has been typical [RODRJ 72]. Further, the number of 'batch equivalents' in interactive systems has been observed to fall in this range since an average batch job constitutes approximately the same system load as ten average interactive jobs [MOORC 71]. Clearly, such estimators are at best crude. Moreover, data on batch equivalents for non-university environments is not readily available although similar computational approaches can be used.

Because of the unsuitability of existing I/O device models, a technique described in [LAVES 75] for obtaining the steady state queuing time distribution for the M/G/1 finite capacity queue proved appropriate and is summarized in the following theorem:

Theorem 1

The expected system residence time of an arriving request to an M/G/1 queue with finite capacity N is given by:

$$* \qquad E[SRT] = NE[S] - \Sigma_{k=1}^{N-1} \; k \; p(k,N)/m,$$

where m is the arrival rate (to the infinite capacity system), E[S] is the average service time and $<p(k,N);\ 1 \leq k \leq N>$ are given by:

$$p(k,N) = a(N-k)/A(N-1);\ 1 \leq k \leq N,$$

and the terms a(j), A(j) are determined recursively through:

$$a(0) = 1$$

$$a(1) = b(1)/(1-b(1))$$

$$a(k) = \Sigma_{u=1}^{k-1} (a(u)b(k+1-u)+a(k))/(1-a(1)), \qquad 2 \le k \le N$$

$$A(k) = \Sigma_{j=0}^{k} \quad a(j); \qquad 0 \le k \le N$$

and

$$b(k) = \int_{0}^{+\infty} (1-F(t))*[(mt)**(k-1)]*m*exp(-mt)/(k-1)! \, dt$$

Note that although data paths are not explicitly incorporated in the present version of ASIM, their incorporation is feasible in the approach described and an approximately linear complexity in terms of the number of devices is retained.

Having obtained the delay for an individual device, average delay for all devices can then be obtained. This then permits determination of processor utilization U and the processor delay D via the following theorem [KIMBS 75]:

Theorem 2

Let A and B denote the average length of the active and blocked intervals of the process. For $K \ge 2$ concurrently executing statistically identical processes:

(i) $U > 1 - \rho^{(K-1)}$

(ii) $D = (K-J)AU(K)$

where $\rho = B/(A+B)$, and $J = B/A$ denotes the expected number of active intervals which can be initiated during a blocked interval.

Application of this theorem and the I/O results described earlier yields all device delays and utilizations. Determination of core utilization is a simple calculation.

ASIM Timing and Implementation Considerations

For portability reasons, ASIM has been implemented in a Fortran superset GASP [PRITA 69]. Although GASP is a simulation language, it was used for its output formatting and report generation capabilities rather than its list manipulation capabilities. Retrospect suggests that direct coding of desired output capabilities and elimination of the GASP routines is feasible and would ease portability.

Description of ASIM speed is misleading since it is a function prima. / of the number of jobs executed during a shift rather than their duration. Thus, execution time is the sum of the time segment execution times which are effectively linear in the number of distinct devices referenced during the time segment. Consequently, overall execution time is nearly linear as a function of the number of jobs and distinct I/O devices. Typically, a time segment performance calcula' on requires approximately one second of processor time on a medium sized computer such as TENEX. Thus, determination of performance statistics for a shift with 50 jobs requires less than two minutes of processor time. Although most shifts process a significantly greater number of jobs, many of these jobs require only very limited amounts of resources and are better represented in the aggregate rather than individually, in view of assumption 3.

Although the description of ASIM calculations is relatively straightforward this reflects hindsight and is not represented in the current version which consists of approximately 1800 source statements. Knuth's remark to the effect that the best way to implement a program is to code it once, throw it away and code it again seems very appropriate. A new version is now being coded and the objective is to reduce the number of lines of source code to approximately 500. The next section provides some comparison information for ASIM evaluation.

AN EXAMPLE

Evaluation of a simulator is a somewhat difficult task. The literature is notably lacking in detailed evaluations and, if discussed at all, evaluation is usually given in terms of a few 'representative' examples. (Evaluation of commercially available simulators is somewhat difficult since most of them require 'tuning'. By contrast, ASIM has virtually no tunable variables.)

Evaluation of ASIM is somewhat simpler since the objective is to demonstrate that information approximately as detailed as that obtained via simulators can be obtained while a significant increase in speed is also obtained. For comparison purposes, a resource allocation simulator (ISIM) [KIMBS 74A] was used. Figures 1-5 provide comparison information. These comparison statistics were obtained for five identical concurrently executing jobs accessing three disks in a uniform manner and differing only in the length of the average processor burst which was 5, 10, 20, 40 and 80 ms. Each job requires 1000 processor bursts, arrived at time 0, had no precedence or due date constraints, and was sufficiently undemanding in core requirements to permit simultaneous initiation.

Figure 1 indicates that the correlation between ASIM and ISIM predictions (for a system whose characteristics are as indicated in Table III) was very good for gross statistics such as the system residence time (elapsed time from initiation to termination of job). Further, Figure 2 shows a reasonable correlation between overall system utilization for ASIM and ISIM as represented in the system reward function (a dollar weighted device utilization statistic computed over all four device categories). Figure 3 demonstrates that the average processor delay also compares well. Figure 4 shows that the comparison between processor utilizations is not as satisfactory as is also true for the comparison between disk delays as shown in Figure 5. This discrepancy reflects the fact that for low access rates to I/O devices, the gap between the computed upper and lower delay bounds for individual devices is fairly wide. Indeed, the envelope between these two bounds contained the ISIM prediction in all cases. Thus, a refinement of the I/O delay prediction technique is required and is perhaps the most important improvement needed. A major requirement for achieving this accuracy improvement is developing an improved estimator for processor utilization, as is apparent from Figure 4.

Collectively, this information indicates that the desired level of accuracy has been achieved for gross overall statistics and that some refinement is needed for the device statistics. Thus, we argue that the basic feasibility of obtaining a 20% accuracy level has been shown and effort directed toward consolidation and improvement is merited. It is natural to consider the quality of the comparisons for non statistically identical jobs. Surprisingly, the comparisons were slightly better and, in general, the accuracy of the predictions seems to improve with increasing system complexity.

DIRECTIONS FOR FUTURE RESEARCH

Results obtained to date support the hypothesis that analytically driven performance prediction techniques providing significant speed advantages over those obtainable via a simulator can be constructed. Further, the discrepancies between ASIM and ISIM appear due to inadequacies in existing analytic techniques rather than to fundamental deficiencies in the approach. Extensive exploration of the limits of this approach is clearly desirable.

Discussion of ASIM speed is misleading since it is effectively independent of the length of the job. For production batch environments in which the average job requires five minutes or more to execute, a claim of two orders of magnitude faster than real time is reasonable. For a university environment in which the average job requires two seconds of processor time this claim is clearly inappropriate; thus the restriction to pro .ction batch environments.

Usage of ASIM is not restricted to batch environments provided that one is willing to represent the interactive load through an 'average' approach. Because the standard deviation of resource requests for interactive jobs is significantly larger than the mean [MOORC 71], development of upper and lower bounds seems more appropriate.

Device utilizations and delays are the usual first order statistics of a stochastic process. In performance analysis second order statistics such as variances and correlations are also known to be useful as demonstrated by the widespread interest in the CPU/channel overlap statistic. Determination of the feasibility of developing such statistics is a research issue and should only be approached after channel effects have been adequately incorporated.

The possibility of very fast performance prediction techniques with information output compatible with that provided by simulations permits essentially different approaches to two important issues.

1. Computer system scheduling is usually performed on a trial and error basis in view of the failure of existing probabilistic and deterministic techniques to adequately reflect the complexity of the computer operation. Thus, probabilistic techniques cannot handle the dimensionality of the system as reflected in both the number of devices and the constraints which must be observed, while deterministic (mathematical programming) techniques cannot reflect the random phenomena occuring in computer systems. Collectively, these factors indicate the desirability of a heuristic approach to scheduling. Using ASIM, an initial feasibility study of this possibility has been undertaken [KIMBS 74B] and the results are encouraging.

2. Design of large networked systems is difficult and currently unsupported by a methodology to reflect the effects of different design

alternatives. Extension of the ASIM approach seems to permit development of such a methodology [KIMBS 74C].

Finally, it is of interest to note that automatic gathering of ASIM input data is feasible with minor modifications to the accounting system on most large computer systems [EDGEG 73]. Thus, the initial requirement for development of an integrated, heuristic approach to data gathering, data transformation through performance prediction and data display has been satisfied. For organizations possessing large collections of geographically dispersed homogeneous computer systems, this potentially permits implementation of a centralized approach to performance analysis and system design.

REFERENCES

BARLR 67 Barlow, R. E., and F. Proschan, ***Mathematical Theory of Reliability***, John Wiley Sons, Inc., New York, 1967.

BASKF 71 Baskett, F., "The Dependence of Computer System Queues Upon Processing Time Distribution and Central Processor Scheduling," ***Proceedings of the Third Symposium on Operating Systems Principles***, October 1971, pp. 109-113.

BUCHW 69 Buchholz, W., "A Synthetic Job for Measuring System Performance," ***IBM Systems Journal***, Vol. 8, No. 4 (1969), pp. 309-318.

BUTLM 74 Butler, M. K. and A. Hirsch, ***Use of the Argonne Benchmark Collection to Measure Computer System Performance***, Argonne National Laboratory, Applied Matnematics Division, ANL-8127, September 1974.

BUZEJ 71 Buzen, J., "Analysis of System Bottlenecks Using a Queuing Network Model," ***Proceedings of ACM (SIGOPS) Workshop on System Performance Evaluation***, Harvard University, April 5-7, 1971, pp. 82-103.

BUZEJ 73 --- "Computational Algorithms for Closed Queuing Networks with Experimental Servers," ***Communications of the ACM***, September 1973, Vol. 16, No. 9, pp. 527-531.

COUGJ 73 Cougar, J. D., "Evolution of Business System Analysis Techniques," ***Computing Surveys***, Vol. 5, No. 3, September 1973, pp. 167-198.

EDGEG 73 Edgecomb, G., "SCERT Model Building System. User's Guide," Chemical Abstracts Service, Columbus, Ohio, September 1973, PB 234 934.

GAVED 67 Gaver, D. P., "Probability Models for Multiprogramming Computer Systems," ***Journal of the ACM***, Vol. 14, No. 3, July 1967, pp. 423-438.

HAMIP 73 Hamilton, P. A., and B. W. Kernigan, "Synthetically Generated Performance Test Loads for Operating Systems," ***Proceedings of the ACM-SIGME Symposium on Measurement and Evaluation***, February 26-28, 1973, pp. 121-126.

HANSF 71 Hanssmann, F., W. Kistler, and H. Schulz, "Modeling for Computer Center Planning," ***IBM Systems Journal***, Vol. 10, No. 4, 1971, pp. 305-324.

KIMBS 72 Kimbleton, S. R., "Performance Evaluation--A Structured Approach," *AFIPS Conference Proceedings*, 1972 Spring Joint Computer Conference, pp. 411-416.

KIMBS 74A --- "Interrupt-Free Computer System Simulator (ISIM) Description," USC/Information Sciences Institute Internal Document, 1974.

KIMBS 74B --- "Batch Computer Scheduling: A Heuristically Motivated Approach," *Proceedings of the Second Annual SIGMETRICS Symposium on Measurement and Evaluation*, Montreal, Canada, September 30--October 2, 1974.

KIMBS 74C --- "Modeling Considerations in Computer Communication Resource Control," *Proceedings of the Eighth Hawaii International Conference on System Sciences*, January 7-9, 1975, Honolulu, Hawaii, pp. 95-97.

KIMBS 75 --- *An Approximate Analytic Technique for Hierarchical Computer System Modeling*, ISI/RR-75-30, to appear in April, 1975.

KLEIL 64 Kleinrock, L., *Communication Nets*, McGraw-Hill Book Company, Inc., New York, 1964.

KLEIL 70 --- "Analytic and Simulation Methods in Computer Network Design," *AFIPS Conference Proceedings*, 1970 Spring Joint Computer Conference, pp. 569-579.

KOBAH 74 Kobayashi, H., "Application of the Diffusion Approximation to Queuing Networks I: Equilibrium Queue Distributions," *Journal of the ACM*, Vol. 21, No. 2, 1974, pp. 316-328.

LAVES 75 Lavenberg, S. S., "The Steady-State Queuing Time Distribution for the M/G/I Finite Capacity Queue," *Management Science*, Theory Series, Vol. 21, No. 5, January 1975, pp. 501-506.

LUCAH 71 Lucas, H. C., "Performance Evaluation and Monitoring," *Computing Surveys*, Vol. 3, No. 3, September, 1971, pp. 79-92.

MOORC 71 Moore, Charles G., *Network Models for Large-Scale Time-Sharing Systems*, Technical Report No. 71-1, Department of Industrial Engineering, The University of Michigan, Ann Arbor, Michigan, 1971.

MUNTR 74 Muntz, R. R. and J. W. Wong, "Efficient Computational Procedures for Closed Queuing Network Models," *Proceedings of the Seventh Hawaii International Conference on System Sciences*, University of Hawaii, Honolulu, 1974.

PRITA 69 Pritsker, A.A.B., and P. J. Kiviat, *Simulation with GASP II*, Prentice-Hall, Inc., Englewood Cliffs, New Jersey, 1969.

RODRJ 72 Rodriguez-Rosell, J. and J. Dupuy, "Instrumentation for the Evaluation of a Time-Sharing, Page Demand System," *AFIPS Conference Proceedings*, 1972 Spring Joint Computer Conference, pp. 759-766.

SEKIA 72 Sekino, A., *Performance Evaluation of a Multiprogrammed Time-Shared Computer System*, Ph.D. Thesis, MIT-Lincoln Laboratory, August 1972.

SREEK 74 Sreenivasan, K., and A. J. Kleinman, "On the Construction of a Representative Synthetic Workload," *Communications of the ACM*, Vol. 27, No. 3, 1974, pp. 127-132.

TEICD 71 Teichroew, D. and H. Sayani, "Automation of System Building," *Datamation*, August 15, 1971, pp. 25-30.

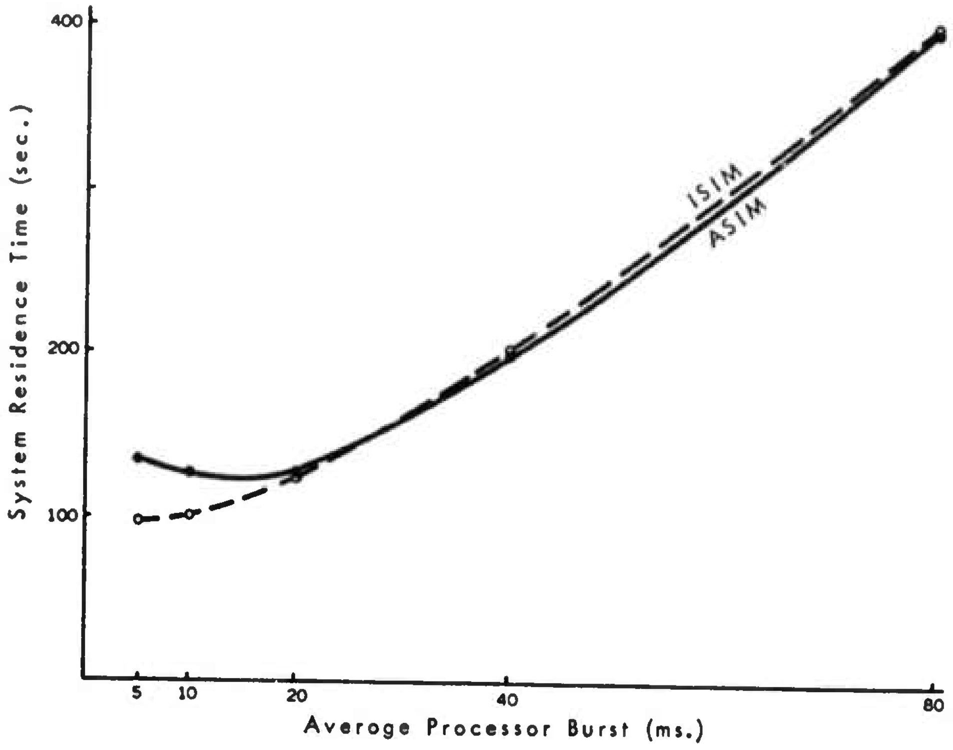

Figure 1. ASIM-ISIM system residence time comporison.

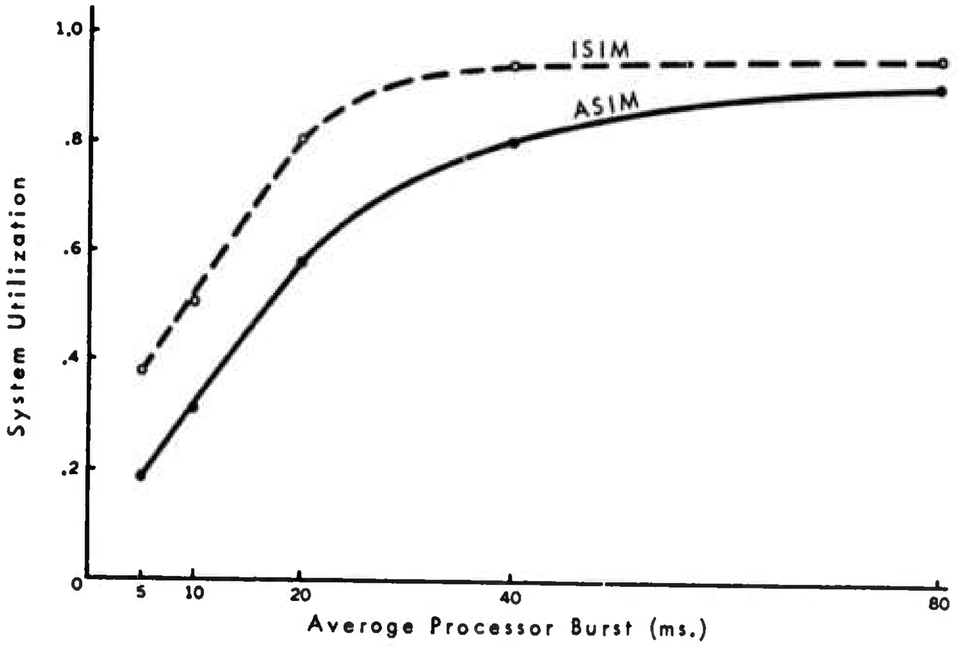

Figure 2. ASIM-ISIM cost weighted system utilizotion comparison.

Preceding page blank

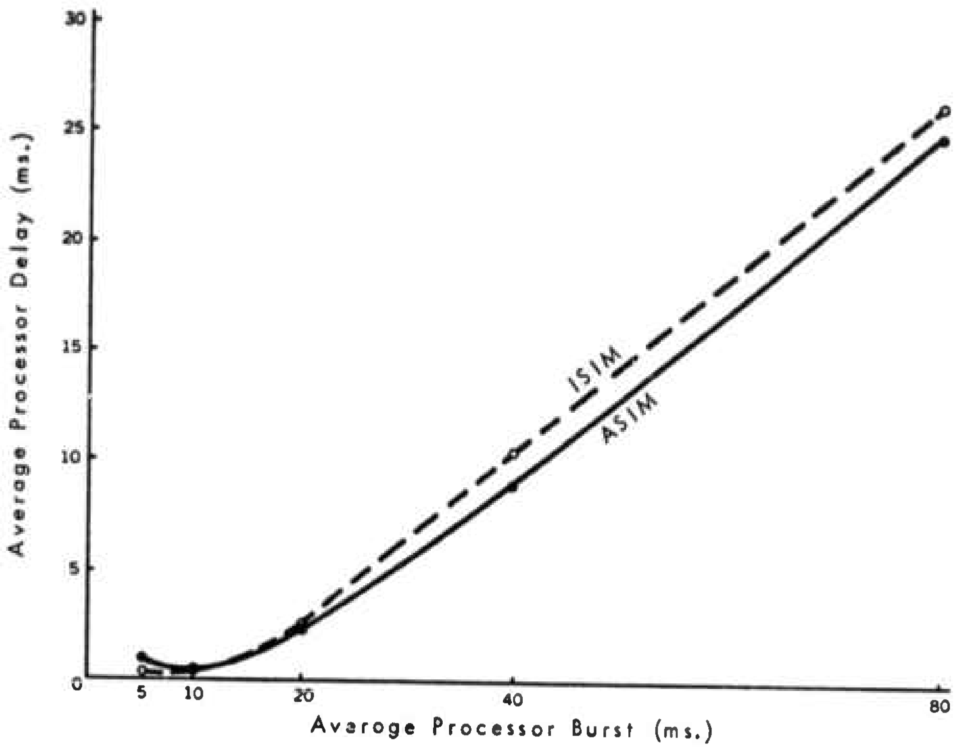

Figure 3. ASIM-ISIM processor deloy comporison.

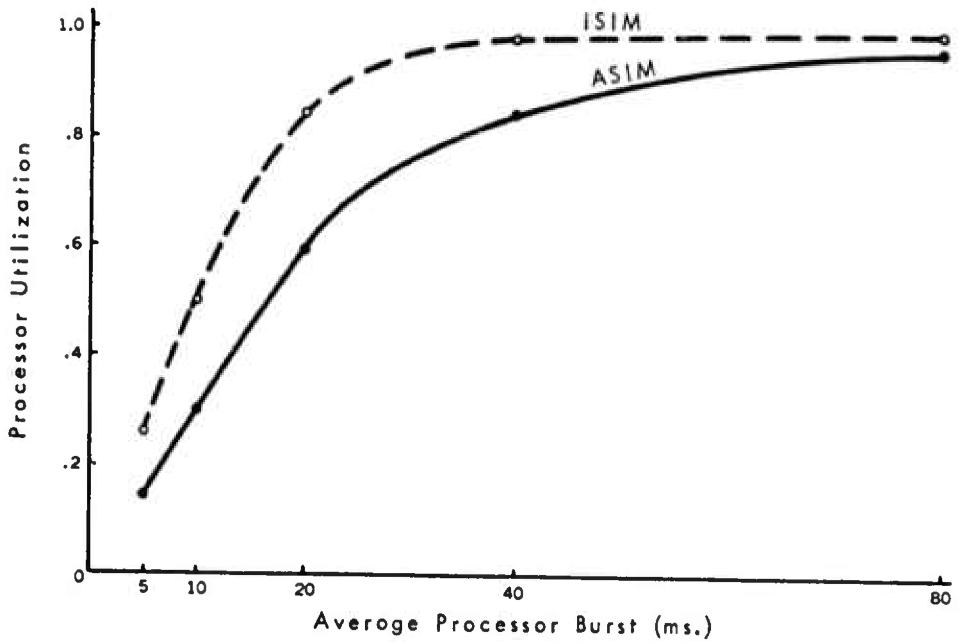

Figure 4. ASIM-ISIM processor utilizotion comporison.

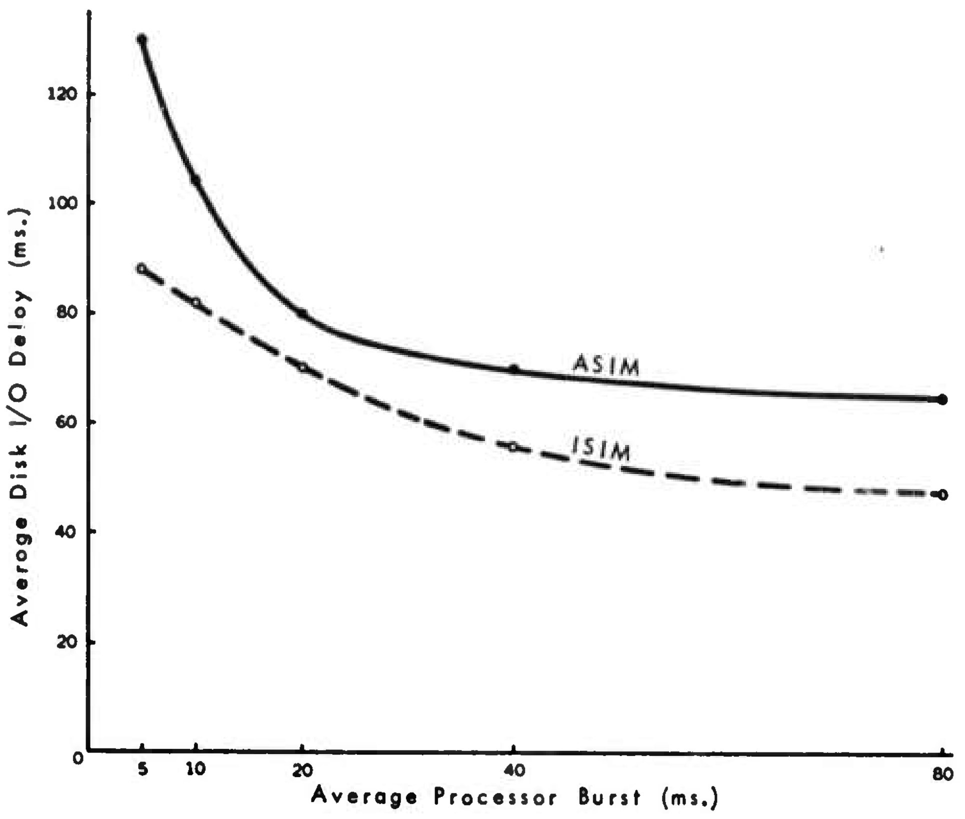

Figure 5. ASIM-ISIM disk I/O delay comparison.